Ref #	Hits	Search Query	DBs	Default Operator	Plurals	Time Stamp
L4	0	(717/126.ccls.) and (generic same customiz$6 same central)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/09/01 13:41
L3	0	l2 and (generic same customiz$6 same central)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/09/01 13:40
L2	223	717/125.ccls. Scan all	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/09/01 13:40
S136	60	generic same customiz$6 same central Rev adj	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/09/01 13:39
L1	1	(717/124-126.ccls.) and (generic same customiz$6 same central)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/09/01 13:39
S134	2596	generic same customiz$6 Scan	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/09/01 10:19
S133	48	generic same (web adj page) same customiz$6 Rev adj	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/09/01 10:18
S132	1	"6934934".pn.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/09/01 08:28
S1	24	central$8 near2 content near2 management	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/09/01 08:28

S131	2	"6021433".pn. Rev all	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/08/31 14:19
S130	266	remote adj debug$6 Scan all	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/08/25 13:37
S129	153	central$6 same content near3 manag$6 and (test$4)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/08/24 17:34
S128	295	central$6 same content near3 manag$6 and (customiz$6 or test$4)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/08/24 17:33
S127	236	central$6 same content near3 manag$6 and customiz$6	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/08/24 17:32
S126	23	central$6 same content near3 manag$6 and debug$4 Rev all	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/08/24 16:57
S125	541	central$6 same content near3 manag$6 Scan	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB	OR	OFF	2005/08/24 15:13

Terms used remote debugging

Found 14,438 of 160,906

Sort results by relevance

Display results expanded form

Save results to a Binder

Search Tips

Open results in a new window

Try an Advanced Search

Try this search in The ACM Guide

Results 1 - 20 of 200

Result page: 1 2 3 4 5 6 7 8 9 10 next

Best 200 shown

Relevance scale

1 Debugging heterogeneous distributed systems using event-based models of behavior

Peter Bates

November 1988 **ACM SIGPLAN Notices , Proceedings of the 1988 ACM SIGPLAN and SIGOPS workshop on Parallel and distributed debugging**, Volume 24 Issue 1

Full text available: pdf(1.52 MB) Additional information: full citation, abstract, references, citings, index terms

Event Based Behavioral Abstraction (EBBA) is a high-level debugging approach which treats debugging as a process of creating models of actual behavior from the activity of the system and comparing these to models of expected system behavior. The differences between the actual and expected models are used to characterize erroneous system behavior and direct further investigation. A set of EBBA-based tools has been implemented that ...

2 Selective interpretation as a technique for debugging computationally intensive programs

B. B. Chase, R. T. Hood

July 1987 **ACM SIGPLAN Notices , Papers of the Symposium on Interpreters and interpretive techniques**, Volume 22 Issue 7

Full text available: pdf(772.68 KB) Additional information: full citation, abstract, index terms

As part of Rice University's project to build a programming environment for scientific software, we have built a facility for program execution that solves some of the problems inherent in debugging large, computationally intensive programs. By their very nature such programs do not lend themselves to full-scale interpretation. In moderation however, interpretation can be extremely useful during the debugging process. In addition to discussing the particular benefits that we expect from interpre ...

3 Debugging distributed object applications with the Eclipse platform

Giuliano Mega, Fabio Kon

October 2004 **Proceedings of the 2004 OOPSLA workshop on eclipse technology eXchange**

Full text available: pdf(244.99 KB) Additional information: full citation, abstract, references

Debugging distributed applications is a well known challenge within the realm of Computer Science. Common problems faced by developers include: lack of an observable global state, lack of a central location from where to monitor possible states, non-deterministic execution, heisenbugs, and many others. There are currently many good techniques available which could be employed in building a tool for circumventing some of those issues, especially when considering wide-spread middleware-induced mod ...

4 OS Debugging Method Using a Lightweight Virtual Machine Monitor

Tadashi Takeuchi

March 2005 **Proceedings of the conference on Design, Automation and Test in Europe - Volume 2**

Full text available: pdf(85.92 KB) Additional information: full citation, abstract

Demands for implementing original OSs that can achieve high I/O performance on PC/AT compatible hardware have recently been increasing, but conventional OS debugging environments have not been able to simultaneously assure their stability, be easily customized to new OSs and new I/O devices, and assure efficient execution of I/O operations. We therefore developed a novel OS debugging method using a lightweight virtual machine. We evaluated this debugging method experimentally and confirmed that ...

5 Networks and distributed systems: Supporting interactive invocation of remote services within an integrated programming environment

Bruce Quig, John Rosenberg, Michael Kölling

June 2003 **Proceedings of the 2nd international conference on Principles and practice of programming in Java PPPJ '03**

Full text available: pdf(360.62 KB) Additional information: full citation, abstract, references, index terms

Building distributed systems is an inherently difficult and complex task. Modern middleware

architectures assist developers by providing abstractions that hide transport layer functionality. This paper argues that the development of such systems can be aided by the availability of appropriate, integrated tools. We discuss ways in which the building of such systems can be supported by development tools, focusing particularly on interactive testing and debugging mechanisms.A prototype system based ...

Keywords: BlueJ, debugging, distributed Systems, testing

6 Debugging heterogeneous distributed systems using event-based models of behavior
Peter C. Bates
February 1995 **ACM Transactions on Computer Systems (TOCS)**, Volume 13 Issue 1
Full text available: pdf(2.15 MB) Additional information: full citation, abstract, references, citings, index terms, review

We describe a high-level debugging approach, Event-Based Behavioral Abstraction (EBBA), in which debugging is treated as a process of creating models of expected program behaviors and comparing these to the actual behaviors exhibited by the program. The use of EBBA techniques can enhance debugging-tool transparency, reduce latency and uncertainty for fundamental debugging activities, and accommodate diverse, heterogeneous architectures. Using events and behavior models as a basic mechanism ...

Keywords: behavior modeling, debugging, events

7 Experience with topaz telebugging
David D. Redell
November 1988 **ACM SIGPLAN Notices , Proceedings of the 1988 ACM SIGPLAN and SIGOPS workshop on Parallel and distributed debugging**, Volume 24 Issue 1
Full text available: pdf(1.06 MB) Additional information: full citation, abstract, references, citings, index terms

The Topaz TeleDebug (TTD) facility provides a remote debugging capability supporting software development in the Topaz environment. Topaz is a software environment providing rich support for programming in Modula2+, and extended version of Modula 2. TTD allows uniform use of the same high level source language debugger for all debugging (both local and remote) of software at all levels of the system. Special care has been taken to maximize TTD's reliability and robustness. Our experience su ...

8 Distributed systems - programming and management: On remote procedure call
Patrícia Gomes Soares
November 1992 **Proceedings of the 1992 conference of the Centre for Advanced Studies on Collaborative research - Volume 2**
Full text available: pdf(4.52 MB) Additional information: full citation, abstract, references, citings

The Remote Procedure Call (RPC) paradigm is reviewed. The concept is described, along with the backbone structure of the mechanisms that support it. An overview of works in supporting these mechanisms is discussed. Extensions to the paradigm that have been proposed to enlarge its suitability, are studied. The main contributions of this paper are a standard view and classification of RPC mechanisms according to different perspectives, and a snapshot of the paradigm in use today and of goals for t ...

9 A structural view of the Cedar programming environment
Daniel C. Swinehart, Polle T. Zellweger, Richard J. Beach, Robert B. Hagmann
August 1986 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 8 Issue 4
Full text available: pdf(6.32 MB) Additional information: full citation, abstract, references, citings, index terms

This paper presents an overview of the Cedar programming environment, focusing on its overall structure—that is, the major components of Cedar and the way they are organized. Cedar supports the development of programs written in a single programming language, also called Cedar. Its primary purpose is to increase the productivity of programmers whose activities include experimental programming and the development of prototype software systems for a high-performance personal computer. T ...

10 Session 24: software tools: A portable debugger for parallel and distributed programs
Doreen Cheng, Robert Hood
November 1994 **Proceedings of the 1994 ACM/IEEE conference on Supercomputing**
Full text available: pdf(996.90 KB) Additional information: full citation, abstract, references, citings

We describe the design and implementation of a portable debugger for parallel and distributed programs. The design incorporates a client-server model in order to isolate non-portable debugger code from the user interface. The precise definition of a protocol for client-server interaction facilitates a high degree of client portability. Replication of server components permits the implementation of a debugger for distributed computations.Portability across message passing implementations is achie ...

11 Interactive blackbox debugging for concurrent languages
G. Goldszmidt, S. Katz, S. Yemini
November 1988 **ACM SIGPLAN Notices , Proceedings of the 1988 ACM SIGPLAN and SIGOPS workshop on Parallel and distributed debugging**, Volume 24 Issue 1
Full text available: Additional information:

pdf(1.31 MB) full citation, abstract, references, citings, index terms

We describe a novel approach to the design of portable integrated debugging tools for concurrent languages. Our design partitions the tools set into two categories. The language specific tools take into account the particular features of a programming language for on-line experimenting with and monitoring of distributed programs. The language independent tools support off-line presentation and analysis of the monitored information. The separation of the lan ...

[12] Generalized path expressions: a high level debugging mechanism
Bernd Bruegge, Peter Hibbard
March 1983 **Proceedings of the symposium on High-level debugging**, Volume 8 , 18 Issue 4 , 8
Full text available: pdf(921.72 KB) Additional information: full citation, abstract, references

This paper introduces a modified version of path expressions called *Path Rules* which can be used as a debugging mechanism to monitor the dynamic behaviour of a computation. Path rules have been implemented in a remote symbolic debugger running on the Three Rivers Computer Corporation PERQ computer under the Accent operating system.

[13] Standardization approach of ITRON debugging interface specification and evaluation of its adaptability
Takayuki Wakabayashi, Hiroaki Takada
June 2002 **ACM SIGPLAN Notices , Proceedings of the joint conference on Languages, compilers and tools for embedded systems: software and compilers for embedded systems**, Volume 37 Issue 7
Full text available: pdf(180.57 KB) Additional information: full citation, abstract, references, index terms

Debugging environments for embedded systems unavoidably depend on the internal structure of the operating system (OS) in order to implement OS support functions. Since the ITRON specification standardizes only the API, the internal structure of operating systems conforming to the ITRON Specification are different, resulting in difficulties in supporting ITRON-Specification operating systems for debugging environments. To solve this problem, we design the ITRON Debugging Interface Specification w ...

Keywords: ITRON specification, OS-aware debugging environment, cross development system environment

[14] Remote evaluation
James W. Stamos, David K. Gifford
October 1990 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 12 Issue 4
Full text available: pdf(2.52 MB) Additional information: full citation, abstract, references, citings, index terms, review

A new technique for computer-to-computer communication is presented that can increase the performance of distributed systems. This technique, called remote evaluation, lets one computer send another computer a request in the form of a program. A computer that receives such a request executes the program in the request and returns the results to the sending computer. Remote evaluation provides a new degree of flexibility in the design of distributed systems. In present distributed systems th ...

[15] Codebugger: a software tool for cooperative debugging
Gaoyan Xie, Yongsen Xu, Yu Li, Qian Li
February 2000 **ACM SIGPLAN Notices**, Volume 35 Issue 2
Full text available: pdf(619.35 KB) Additional information: full citation, abstract, index terms

Debugging of large-scale systems are essentially cooperative and need collaboration of group developers. However, most existing debugging tools are for single developer. With single-user tools, only one developer can operate the debugging process while others are frequently requested to gather before the screen of a single computer on solving problems, which makes error location and correction so inconvenient and inefficient. We developed Codebugger---a cooperative debugging tool for Java that s ...

Keywords: CSCW, Jdb, cooperative debugging, debugger

[16] Relative debugging: a new methodology for debugging scientific applications
David Abramson, Ian Foster, John Michalakes, Rok Sosič
November 1996 **Communications of the ACM**, Volume 39 Issue 11
Full text available: pdf(462.99 KB) Additional information: full citation, references, citings, index terms

[17] CAPS: a coding aid for PASM
James E. Lumpp, Samuel A. Fineberg, Thomas L. Casavant, Wayne G. Nation, Edward C. Bronson, Howard Jay Siegel, Pierre H. Pero, Dan C. Marinescu, Thomas Schwederski
November 1991 **Communications of the ACM**, Volume 34 Issue 11
Full text available: Additional information:

pdf(5.80 MB) full citation, abstract, references, citings, index terms, review

Programming parallel machines is very difficult. First, generating an algorithm requires the programmer to assimilate the interactions of multiple threads of control. Second, synchronization and communication among the threads must be addressed to avoid contention and deadlock. Then, once the program is executing on the parallel system and does not function correctly or performs poorly, the debugging of multiple threads is a complicated problem [21]. Additionally, debugging software is an a ...

Keywords: instrumentation

18 The langley research center remote computing terminal system: Implementation and first year's operation

Roger V. Butler

January 1966 **Proceedings of the 1966 21st national conference**

Full text available: pdf(1.63 MB) Additional information: full citation, abstract, references, index terms

In April 1965 a remote computing terminal system was installed at the Langley Research Center in support of open shop programming. This paper describes the system and relates some of the experience gained in using and managing it. Because this system grew out of the needs of the open shop programmers, I will begin by reviewing the development of our open shop programming project. This work got under way in 1957 when it was decided to train some of the research scientists at the Center to us ...

19 The structure of Cedar

Daniel C. Swinehart, Polle T. Zellweger, Robert B. Hagmann

June 1985 **Proceedings of the ACM SIGPLAN 85 symposium on Language issues in programming environments**, Volume 20 , 18 Issue 7 , 6

Full text available: pdf(1.79 MB) Additional information: full citation, abstract, references, citings, index terms

This paper presents an overview of the Cedar programming environment, focusing primarily on its overall structure: the major components of Cedar and the way they are organized. Cedar supports the development of programs written in a single programming language, also called Cedar. We will emphasize the extent to which the Cedar language, with runtime support, has influenced the organization, comprehensibility, and stability of Cedar. Produced in the Computer Science Laboratory (CS ...

20 Hardware assisted high level debugging: preliminary draft

W. Morven Gentleman, Henry Hoeksma

March 1983 **Proceedings of the symposium on High-level debugging**, Volume 8 , 18 Issue 4 , 8

Full text available: pdf(489.23 KB) Additional information: full citation, abstract, references

Hardware assistance has long been used for logic level and functional unit level hardware debugging, as well as for machine language level software debugging. Such hardware assistance includes probes to detect signals, comparators to identify matches with expected patterns, buffers to record selected events, and independent logic and software to analyze and interpret the observed events. It can also include the ability to generate selected signals to stimulate the object being debugged and the a ...

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next